

Marco-o1: Towards Open Reasoning Models for Open-Ended Solutions

Yu Zhao^{*}, Huifeng Yin^{*}, Bo Zeng, Hao Wang, Tianqi Shi, Chenyang Lyu, Longyue Wang, Weihua Luo and Kaifu Zhang

MarcoPolo Team, Alibaba International Digital Commerce

Currently OpenAI o1 has sparked a surge of interest in the study of large reasoning models (LRM). Building on this momentum, Marco-o1 not only focuses on disciplines with standard answers, such as mathematics, physics, and coding—which are well-suited for reinforcement learning (RL)—but also places greater emphasis on open-ended resolutions. We aim to address the question: “Can the o1 model effectively generalize to broader domains where clear standards are absent and rewards are challenging to quantify?” Marco-o1 is powered by Chain-of-Thought (CoT) fine-tuning, Monte Carlo Tree Search (MCTS), reflection mechanisms, and innovative reasoning strategies—optimized for complex real-world problem-solving tasks.

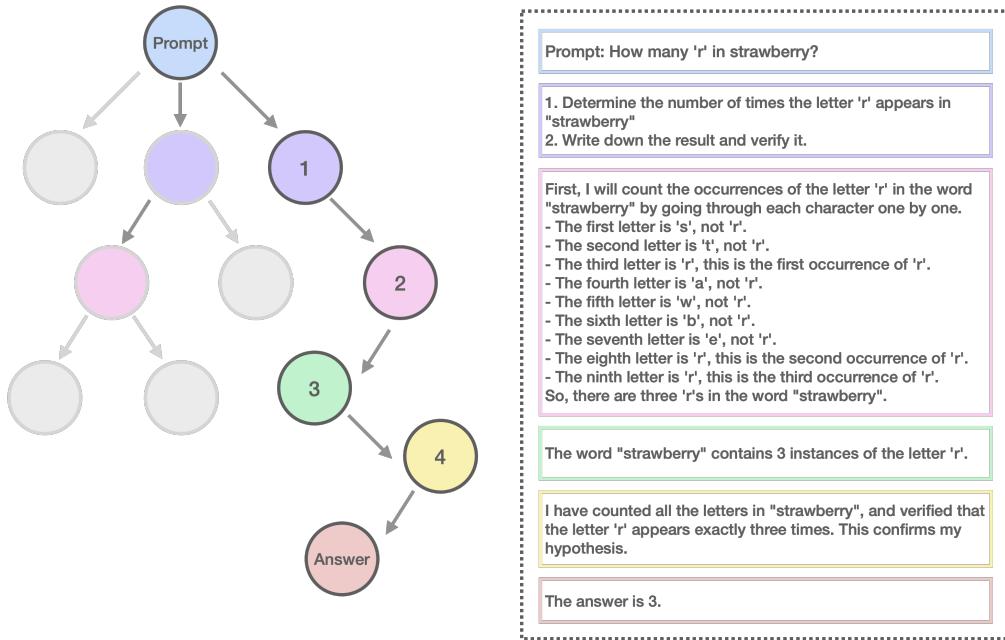


Figure 1 | A classic ‘strawberry’ question reasoned by our Marco-o1 model: “How many ‘r’s are in ‘strawberry’.”

1. Introduction

OpenAI recently introduced the groundbreaking o1 model [OpenAI, 2024, Zhong et al., 2024], renowned for its exceptional reasoning capabilities. This model has demonstrated outstanding performance on platforms such as AIME and CodeForces, surpassing other leading models. Inspired by this success, we aimed to push the boundaries of LLMs even further, enhancing their reasoning abilities to tackle complex, real-world challenges.

^{*}Equal Contribution.

[†]Email: wanglongyue.wly@alibaba-inc.com

Marco-o1 leverages advanced techniques like CoT fine-tuning [Wei et al., 2022], MCTS [Wei et al., 2022, Feng et al., 2023, Silver et al., 2017], and Reasoning Action Strategies to enhance its reasoning power. As shown in Figure 2, by fine-tuning Qwen2-7B-Instruct [Yang et al., 2024] with a combination of the filtered Open-O1 CoT dataset [Team, 2024], Marco-o1 CoT dataset, and Marco-o1 Instruction dataset, Marco-o1 improved its handling of complex tasks. MCTS allows exploration of multiple reasoning paths using confidence scores derived from softmax-applied log probabilities of the top- k alternative tokens, guiding the model to optimal solutions. Moreover, our reasoning action strategy involves varying the granularity of actions within steps and mini-steps to optimize search efficiency and accuracy.

Marco-o1 achieved accuracy improvements of +6.17% on the MGSM (English) dataset and +5.60% on the MGSM (Chinese) dataset, showcasing enhanced reasoning capabilities [Shi et al., 2022]. Additionally, in translation tasks, we demonstrate that Marco-o1 excels in translating slang expressions. For example, the model correctly translates a colloquial expression in Chinese that literally means “This shoe offers a stepping-on-poop sensation” to English “This shoe has a comfortable sole,” demonstrating its superior grasp of colloquial nuances.

Our work is distinguished by the following contributions:

- **Fine-Tuning with CoT Data:** We develop Marco-o1-CoT by performing full-parameter fine-tuning on the base model using open-source CoT datasets combined with our self-developed synthetic data.
- **Solution Space Expansion via MCTS:** We integrate LLMs with MCTS (Marco-o1-MCTS), using the model’s output confidence to guide the search and expand the solution space.
- **Reasoning Action Strategy:** We implement novel reasoning action strategies and a reflection mechanism (Marco-o1-MCTS mini-step), including exploring different action granularities within the MCTS framework and prompting the model to self-reflect, thereby significantly enhancing the model’s ability to solve complex problems.
- **Application in Translation Tasks:** We are the first to apply Large Reasoning Models (LRM) to Machine Translation tasks, exploring inference-time scaling laws in the multilingual and translation domain.

2. Marco Reasoning Datasets

To enhance the reasoning capabilities of the Marco-o1 model, we employed a Supervised Fine-Tuning (SFT) strategy using a variety of datasets.

- **Open-O1 CoT Dataset (Filtered) [Team, 2024]:** We refined the Open-O1 project’s CoT Dataset by applying heuristic and quality filtering processes. This enhancement allowed the model to adopt structured reasoning patterns effectively.
- **Marco-o1 CoT Dataset (Synthetic):** We generated the Marco-o1 CoT Dataset using MCTS, which helped to formulate complex reasoning pathways, further bolstering the model’s reasoning capabilities.
- **Marco Instruction Dataset:** Recognizing the critical role of robust instruction-following capabilities in executing complex tasks, we incorporated a set of instruction-following data. This integration ensures the model remains competent across a wide range of tasks, maintaining its general effectiveness while significantly boosting its reasoning flair.

*Download: [Marco Reasoning Dataset](#) (Our Partial Dataset).

Dataset	Number of Samples
Open-O1 CoT Dataset (Filtered) [Team, 2024]	45,125
Marco-o1 CoT Dataset (Synthetic)	10,000
Marco Instruction Dataset	5,141
Total	60,266

Table 1 | Overview of Marco Reasoning Datasets

3. Solution Space Expansion via MCTS

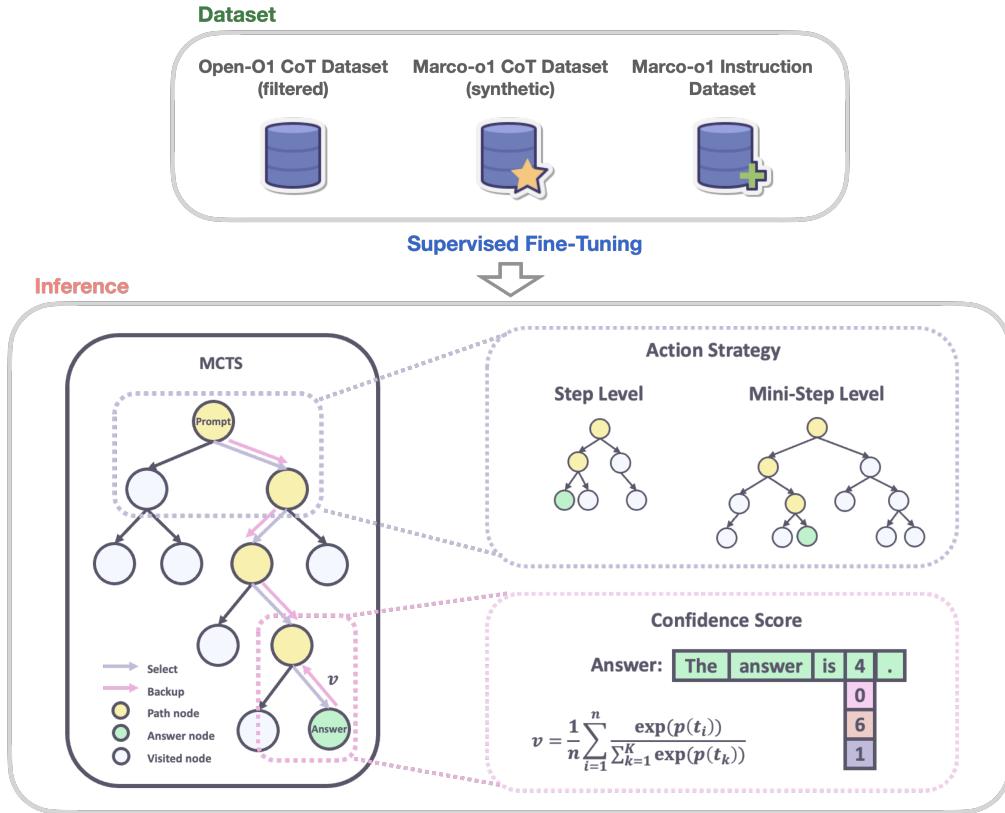


Figure 2 | The overview of Marco-o1.

We integrated LLMs with MCTS to enhance the reasoning capabilities of our Marco-o1 model:

- **Nodes as Reasoning States:** In the MCTS framework, each node represents a reasoning state of the problem-solving process.
- **Actions as LLM Outputs:** The possible actions from a node are the outputs generated by the LLM. These outputs represent potential steps or mini-steps in the reasoning chain.
- **Rollout and Reward Calculation:** During the rollout phase, the LLM continues the reasoning process to a terminal state.
- **Guiding MCTS:** This reward score R is used to evaluate and select promising paths within the MCTS, effectively guiding the search towards more confident and reliable reasoning chains.

Furthermore, we obtain the value of each state by computing a confidence score. For each token t_i generated during the rollout, we calculate its confidence score by applying the softmax function to

its log probability and the log probabilities of the top 5 alternative tokens. This is given by:

$$c_i = \frac{\exp(p(t_i))}{\sum_{k=1}^5 \exp(p(t_k))}$$

where c_i is the confidence score for the i^{th} token in the rollout. $p(t_i)$ is the log probability of the i^{th} token generated by the LLM. $p(t_k)$ for $k = 1$ to 5 are the log probabilities of the top 5 predicted tokens at the i^{th} step. n is the total number of tokens in the rollout sequence. This equation ensures that the confidence score reflects the relative probability of the chosen token compared to the top alternatives, effectively normalizing the scores between 0 and 1.

After obtaining the confidence scores for all tokens in the rollout sequence, we compute the average confidence score across all tokens to derive the overall reward score:

$$\nu = \frac{1}{n} \sum_{i=1}^n c_i$$

where ν is the overall reward score for the rollout path. This average serves as the reward signal that evaluates the quality of the reasoning path taken during the rollout. A higher ν indicates a more confident and likely accurate reasoning path.

By employing this method, we effectively expand the solution space, allowing the model to explore a vast array of reasoning paths and select the most probable ones based on calculated confidence scores.

4. Reasoning Action Strategy

4.1. Action Selection

We observed that using actions as the granularity for MCTS search is relatively coarse, often causing the model to overlook nuanced reasoning paths crucial for solving complex problems. To address this, we explored different levels of granularity in the MCTS search. Initially, we used steps as the unit of search. To further expand the model's search space and enhance its problem-solving capabilities, we experimented with dividing these steps into smaller units of 64 or 32 tokens, referred to as "mini-step." This finer granularity allowed the model to explore reasoning paths in greater detail. While token-level search offers theoretical maximum flexibility and granularity, it is currently impractical due to the significant computational resources required and the challenges associated with designing an effective reward model at this level.

In our experiments, we implemented the following strategies within the MCTS framework:

- **Step as Action:** We allowed the model to generate complete reasoning steps as actions. Each MCTS node represents an entire thought or action label. This method enables efficient exploration but may miss finer-grained reasoning paths essential for complex problem-solving.
- **Mini-step as Action:** We used mini-steps of 32 or 64 tokens as actions. This finer granularity expands the solution space and improves the model's ability to navigate complex reasoning tasks by considering more nuanced steps in the search process. By exploring the solution space at this level, the model is better equipped to find correct answers that might be overlooked with larger action units.

4.2. Reflection after Thinking

We introduced a reflection mechanism by adding the phrase “*Wait! Maybe I made some mistakes! I need to rethink from scratch.*” at the end of each thought process. This prompts the model to self-reflect and reevaluate its reasoning steps. Implementing this reflection has yielded significant improvements, especially on difficult problems that the original model initially solved incorrectly. With the addition of reflection, approximately half of these challenging problems were answered correctly.

From the self-critic perspective [Valmeekam et al., 2023], this approach allows the model to act as its own critic, identifying potential errors in its reasoning. By explicitly prompting the model to question its initial conclusions, we encourage it to re-express and refine its thought process. This self-critical mechanism leverages the model’s capacity to detect inconsistencies or mistakes in its own output, leading to more accurate and reliable problem-solving [Madaan et al., 2024, Li et al., 2024, Huang et al., 2022]. The reflection step serves as an internal feedback loop, enhancing the model’s ability to self-correct without external intervention.

5. Experiments

Based on **Qwen2-7B-Instruct**, we performed SFT using our training data to create **Marco-o1-CoT**. Besides, we employed Marco-o1-CoT within the framework of MCTS tree search, differentiating by actions:

- **Marco-o1-MCTS (step)**: using each inference step as an action (step).
- **Marco-o1-MCTS (mini-step of 64 tokens)**: using a 64-token mini-step as an action (64 tokens).
- **Marco-o1-MCTS (mini-step of 32 tokens)**: using a 32-token mini-step as an action (32 tokens).

During testing, each model utilized a CoT prompt to ensure consistency in reasoning processes. We then tested these configurations on the English (En) and Chinese (Zh) subsets of the MGSM dataset, obtaining the following results:

Model	MGSM-En (Acc.)	MGSM-Zh (Acc.)
Qwen2-7B-Instruct	84.23%	76.80%
Marco-o1-CoT	85.60%	71.20%
Marco-o1-MCTS (step)	90.40%	80.00%
Marco-o1-MCTS (mini-step of 64 tokens)	88.40%	80.40%
Marco-o1-MCTS (mini-step of 32 tokens)	87.60%	82.40%

Table 2 | Experimental results on MGSM datasets

These results demonstrate that:

In the MGSM-en dataset, Marco-o1-CoT shows an advantage over Qwen2-7B-Instruct, as shown in Figure 4, which is expected due to the fine-tuning with English CoT data. In the MGSM-zh dataset, however, Marco-o1-CoT exhibits a decrease in performance compared to Qwen2-7B-Instruct. This decline is attributed to the fact that the CoT data used for fine-tuning was in English, which may not transfer effectively to the Chinese dataset.

The three MCTS-enhanced models demonstrate improvements over Marco-o1-CoT, indicating that incorporating MCTS helps to expand the model’s solution space and increase the probability of obtaining correct answers. However, since we use the Confidence Score as the reward, the tree search

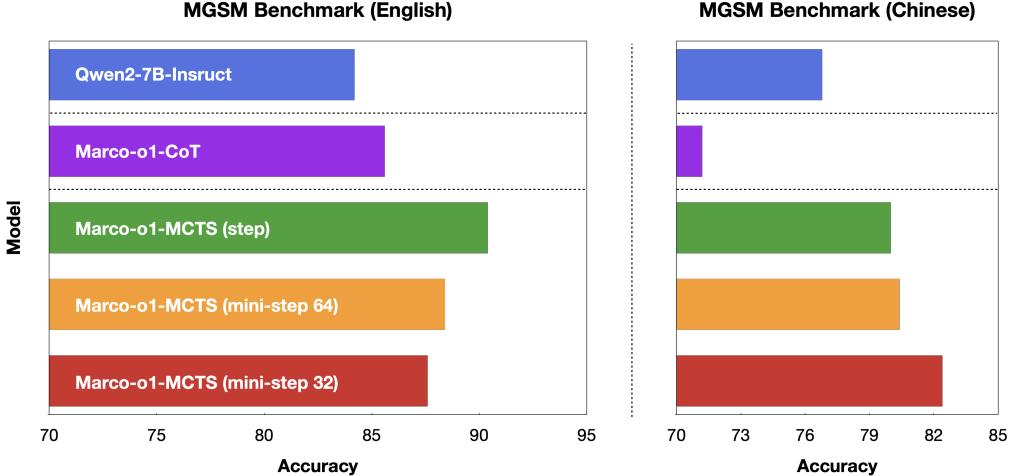


Figure 3 | The main results of Marco-o1.

results exhibit significant randomness. In MGSM-en, the “step as Action” strategy performs the best, while in MGSM-zh, the “mini-step as Action (32)” strategy yields the highest accuracy. Currently, as shown in Figures 4, 5, and 6, we cannot draw definitive conclusions about which action strategy is superior. We believe that as the reward becomes more accurate, the larger solution space provided by MCTS will demonstrate greater potential.

These results demonstrate the effectiveness of our approach in enhancing the reasoning capabilities of the model across different languages and configurations.

6. Case Study on Translation Tasks

To demonstrate the capabilities of our Marco-o1 model in translation tasks, we conducted a case study comparing its performance with Google Translate on translating colloquial and slang expressions. Our model excels in understanding context and nuances, providing more accurate and natural translations.

As shown in Figure 7, 8, and 9, these cases illustrate the advanced understanding and reasoning capabilities of Marco-o1 in handling complex translation tasks, especially with colloquial and slang language, outperforming standard translation tools like Google Translate.

7. Conclusions and Future Work

Our Marco-o1 enhances the reasoning ability by integrating Chain-of-Thought (CoT) fine-tuning, Monte Carlo Tree Search (MCTS), and novel reasoning action strategies. Marco-o1’s integration of MCTS allowed for expanded solution spaces, and experimentation with different action granularities (steps and mini-steps) showed the potential of finer search resolutions in enhancing accuracy. Our approach demonstrated significant improvements in reasoning tasks, as well as success in translating complex slang expressions.

Looking ahead, we aim to refine the reward signal for MCTS through Outcome Reward Modeling (ORM) and Process Reward Modeling (PRM) [Lightman et al., 2023], which will reduce randomness and further improve performance. Additionally, reinforcement learning techniques are being explored to fine-tune the decision-making processes of Marco-o1, ultimately enhancing its ability to tackle complex real-world tasks.

Marco-o1-CoT	Marco-o1-MCTS (step)
<p>Prompt: Carla is downloading a 200 GB file. Normally she can download 2 GB/minute, but 40% of the way through the download, Windows forces a restart to install updates, which takes 20 minutes. Then Carla has to restart the download from the beginning. How long does it take to download the file?</p> <p>CoT:</p> <p>Correct steps: I need to figure out how long it takes Carla to download a 200 GB file under these conditions. Let's break down the problem step by step.\n\n(139 words omitted) Now, let's calculate the time taken for each part: - **First Download Attempt:**\n - Time to download 80 GB: $80 \text{ GB} / 2 \text{ GB/minute} = 40 \text{ minutes}$. - Then, Windows forces a restart, which takes 20 minutes.\n</p> <p>Incorrect steps: - **Second Download Attempt:**\n - Time to download the remaining 120 GB: $120 \text{ GB} / 2 \text{ GB/minute} = 60 \text{ minutes}$.</p> <p>Subsequent incorrect steps: Adding these up:\n- First download: 40 minutes (download) + 20 minutes (restart) = 60 minutes.\n- Second download: 60 minutes.\n\n(207 words omitted) The problem states that the download rate is 2 GB per minute, and the restart is a separate event that doesn't impact the download speed. Therefore, the download rate remains constant throughout.</p> <p>Output: In conclusion, after accounting for the 40% interruption and the 20-minute restart, Carla takes a total of 120 minutes to download the 200 GB file.</p>	<p>Prompt: Carla is downloading a 200 GB file. Normally she can download 2 GB/minute, but 40% of the way through the download, Windows forces a restart to install updates, which takes 20 minutes. Then Carla has to restart the download from the beginning. How long does it take to download the file?</p> <p>CoT:</p> <p>Correct steps: I need to figure out how long it will take Carla to download a 200 GB file under these specific conditions. Let's break down the problem step by step. (106 words omitted) Now, the first interruption: the restart itself takes 20 minutes. After this, Carla has to restart the download from the beginning. This means she needs to redownload the entire 200 GB starting over.</p> <p>Correct steps: Let's summarize the timeline so far: - Downloaded 80 GB (which took $80 \text{ GB} / 2 \text{ GB/min} = 40 \text{ minutes}$) - Restart for 20 minutes - Redownloaded all 200 GB (which takes $200 \text{ GB} / 2 \text{ GB/min} = 100 \text{ minutes}$)</p> <p>Subsequent correct steps: Adding these up gives the total time: 40 minutes + 20 minutes + 100 minutes = 160 minutes. (170 words omitted) In conclusion, taking into account the initial 40% download, the 20-minute restart, and the subsequent full redownload, the total time Carla spends downloading the file is 160 minutes.</p> <p>Output: It takes Carla a total of 160 minutes to download the file.</p>

Figure 4 | MCTS Expands the Solution Space for Correct Answers. Comparison between Marco-o1-CoT (left) and Marco-o1-MCTS (step) (right) on the MGSM dataset. While Marco-o1-CoT failed to provide the correct answer, integrating MCTS with step-level actions allowed the model to explore a broader solution space, increasing the likelihood of arriving at the correct solution.

References

- X. Feng, Z. Wan, M. Wen, S. M. McAleer, Y. Wen, W. Zhang, and J. Wang. Alphazero-like tree-search can guide large language model decoding and training. *arXiv preprint arXiv:2309.17179*, 2023.
- J. Huang, S. S. Gu, L. Hou, Y. Wu, X. Wang, H. Yu, and J. Han. Large language models can self-improve. *arXiv preprint arXiv:2210.11610*, 2022.
- Z. Li, B. Peng, P. He, M. Galley, J. Gao, and X. Yan. Guiding large language models via directional stimulus prompting. *Advances in Neural Information Processing Systems*, 36, 2024.
- H. Lightman, V. Kosaraju, Y. Burda, H. Edwards, B. Baker, T. Lee, J. Leike, J. Schulman, I. Sutskever, and K. Cobbe. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.
- A. Madaan, N. Tandon, P. Gupta, S. Hallinan, L. Gao, S. Wiegreffe, U. Alon, N. Dziri, S. Prabhumoye, Y. Yang, et al. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*, 36, 2024.

Marco-o1-MCTS (step)	Marco-o1-MCTS (mini-step of 32 tokens)
<p>Prompt: Josh decides to try flipping a house. He buys a house for \$80,000 and then puts in \$50,000 in repairs. This increased the value of the house by 150%. How much profit did he make?</p> <p>CoT:</p> <p>Correct steps: I need to figure out how much profit Josh made from his house renovation project. Let's break down the information given: (21 words omitted) First, I'll calculate the total amount Josh spent on the house and repairs. That's \$80,000 plus \$50,000, which equals \$130,000.</p> <p>Incorrect steps: Now, the house's value increased by 150%. To find the new value after the increase, I can multiply the initial investment by 2.5 (since a 100% increase doubles the original amount, and an additional 50% adds half of that). So, \$130,000 multiplied by 2.5 equals \$325,000.</p> <p>Subsequent incorrect steps: To determine the profit, I subtract the total amount invested (\$130,000) from the new value of the house (\$325,000). That gives me \$195,000. (247 words omitted) In conclusion, after carefully analyzing the numbers and verifying through multiple approaches, Josh earned a profit of \$195,000 from his house renovation project.</p> <p>Output: Josh earned a profit of \$195,000 from his house renovation project.</p>	<p>Prompt: Josh decides to try flipping a house. He buys a house for \$80,000 and then puts in \$50,000 in repairs. This increased the value of the house by 150%. How much profit did he make?</p> <p>CoT:</p> <p>Correct steps: I need to figure out how much profit Josh made from his house renovation project. Let's break down the information given: (56 words omitted) - **Total Investment** = Purchase Price + Repair Costs - Total Investment = \$80,000 + \$50,000 = \$130,000 - Now, the house's value increased by 150%. To find out what this increase amounts to in dollars, we'll calculate 150% of the original purchase price.</p> <p>Incorrect steps: - **Value Increase** = 150% of Purchase Price - Value Increase = 1.5 * \$80,000 = \$120,000</p> <p>Subsequent correct steps: Adding this increase to the original purchase price gives us the new value of the house after renovations. (235 words omitted) In conclusion, after carefully analyzing the numbers and verifying through multiple approaches, Josh made a profit of \$70,000 from his house renovation project.</p> <p>Output: Josh earned a profit of \$70,000.</p>

Figure 5 | Finer Granularity with mini-steps Enhances Problem-Solving. Comparison between Marco-o1-MCTS (step) (left) and Marco-o1-MCTS (mini-step of 32 tokens) (right) on the MGSM dataset. The step-level action strategy did not yield the correct answer, but by using a finer-grained mini-step of 32 tokens, the model successfully navigated the solution space to find the correct answer, demonstrating the effectiveness of increased action granularity.

- OpenAI. Learning to reason with llms. <https://openai.com/index/learning-to-reason-with-llms/>, 2024. [Accessed 19-09-2024].
- F. Shi, M. Suzgun, M. Freitag, X. Wang, S. Srivats, S. Vosoughi, H. W. Chung, Y. Tay, S. Ruder, D. Zhou, et al. Language models are multilingual chain-of-thought reasoners. *arXiv preprint arXiv:2210.03057*, 2022.
- D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.
- O. Team. Open-o1. <https://github.com/Open-Source-01/Open-01>, 2024. [Accessed 19-11-2024].
- K. Valmeeekam, M. Marquez, and S. Kambhampati. Can large language models really improve by self-critiquing their own plans? *arXiv preprint arXiv:2310.08118*, 2023.
- J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou, et al. Chain-of-thought

Marco-o1-MCTS (mini-step of 64 tokens)

Prompt: John plans to sell all his toys and use the money to buy video games. He has 13 lego sets and he sells them for \$15 each. He ends up buying 8 video games for \$20 each and has \$5 left. How many lego sets does he still have?

CoT:

Correct steps:

let's tackle this problem step by step to determine how many Lego sets John still has after his transactions.\t\t First, John starts with 13 Lego sets. Each of these Lego sets is sold for \$15. So, the total amount of money he gets from selling the Lego sets can be calculated by multiplying the number of Lego sets by the price per set:13 Lego sets * \$15/set = \$195\t\t

Incorrect steps:

So, John makes \$195 from selling his Lego sets.\t\t Next, John uses this \$195 to buy video games. Each video game costs \$20, and he buys 8 of them. To find out how much money he spends on video games, we multiply the number of video games by the

Subsequent incorrect steps:

(207 words omitted)

Unless there's an implicit action taken with the remaining \$5 that involves acquiring more Lego sets, which isn't specified, John currently has 0 Lego sets left.

Output:

John has 0 Lego sets.

Marco-o1-MCTS (step)

Prompt: John plans to sell all his toys and use the money to buy video games. He has 13 lego sets and he sells them for \$15 each. He ends up buying 8 video games for \$20 each and has \$5 left. How many lego sets does he still have?

CoT:

Correct steps:

let's tackle this problem step by step to determine how many Lego sets John still has after selling some of them and purchasing video games.

John initially has 13 Lego sets. He sells each set for \$15, so he earns $13 * \$15 = \195 from selling the Lego sets. He uses this money to buy 8 video games at \$20 each, which costs him $8 * \$20 = \160 . After these purchases, he has \$5 left.

Incorrect steps:

To find out how many Lego sets he sold, we can subtract the cost of the video games and the remaining money from the total amount earned: $\$195 - \$160 - \$5 = \30 .

Subsequent correct steps:

Since each Lego set was sold for \$15, the number of Lego sets sold is $\$30 / \$15 = 2$.

Therefore, John still has $13 - 2 = **11**$ Lego sets remaining.

Output:

John still has $13 - 2 = **11**$ Lego sets remaining.

Figure 6 | Optimal Action Granularity Depends on Problem Complexity. Comparison between Marco-o1-MCTS (mini-step of 64 tokens) (left) and Marco-o1-MCTS (step) (right) on the MGSM dataset. The model with a mini-step of 64 tokens failed to find the correct answer, whereas using step-level actions enabled the model to correctly solve the problem. This highlights that we cannot draw definitive conclusions about which action strategy is superior. We believe that as the reward becomes more accurate, the larger solution space provided by MCTS will demonstrate greater potential.

prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

A. Yang, B. Yang, B. Hui, B. Zheng, B. Yu, C. Zhou, C. Li, C. Li, D. Liu, F. Huang, et al. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024.

T. Zhong, Z. Liu, Y. Pan, Y. Zhang, Y. Zhou, S. Liang, Z. Wu, Y. Lyu, P. Shu, X. Yu, et al. Evaluation of openai o1: Opportunities and challenges of agi. *arXiv preprint arXiv:2409.18486*, 2024.

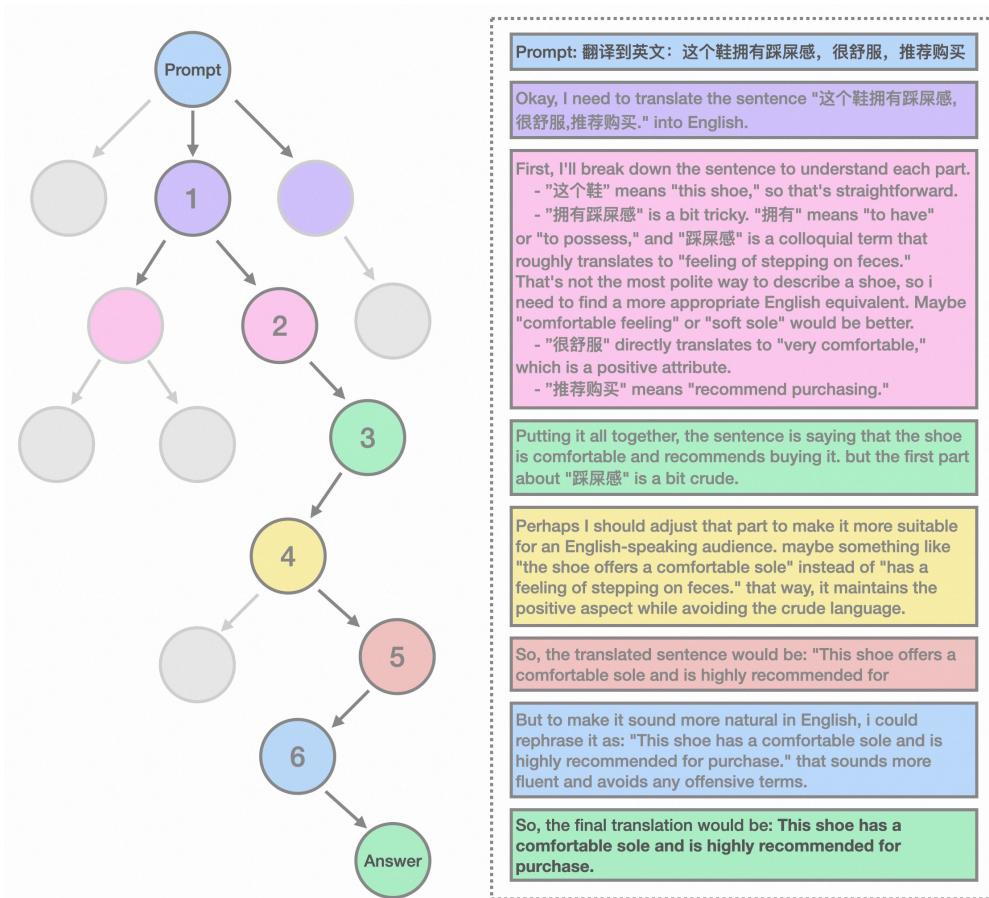


Figure 7 | Demonstration of translation task using Marco-01 of a colloquial expression “This shoe has a comfortable sole and is highly recommended for purchase”.

Input: 美到我心巴上了，上身真的很韩，穿上软乎乎毛茸茸的厚度也刚好，里面搭配了打底衫，小特别且日常的穿搭

Google Translate:

It's so beautiful that I'm attracted to it. The upper body is **really Korean**. It's soft and fluffy and the thickness is just right. It's paired with a base shirt. It's a little special and daily outfit.

Marco-o1-CoT:

ok, I need to translate the given Chinese sentence into English.

First, let's break down the sentence to understand its components:

1. "美到我心巴上了" - This phrase seems to express admiration or being captivated by something, possibly a person or an object.
2. "上身真的很韩" - This indicates that the upper part of **something is very "Korean" in style**.
3. "穿上软乎乎毛茸茸的厚度也刚好" - This describes the texture and thickness of something, likely clothing, as "soft and fluffy" with just the right thickness.
4. "里面搭配了打底衫" - This means that there is a base layer or undershirt underneath.
5. "小特别且日常的穿搭" - This suggests that the overall outfit is both special and casual.

Now, considering the cultural context, "心巴上了" is a colloquial expression that can be translated as "captivated" or "charmed."

....

Alternatively, "It's so beautiful that it's captivating.....

Output:

It's so beautiful that it's captivating, the upper part has a distinctly **Korean style**, the soft and fluffy material is perfectly thick, and it's complemented by a base layer, creating a unique and everyday-wear outfit.

Figure 8 | Translation comparison of a colloquial expression “It's so beautiful that it's captivating, the upper part has a distinctly Korean style, the soft and fluffy material is perfectly thick, and it's complemented by a base layer, creating a unique and everyday-wear outfit”.

Input: 太太太太好看了！而且价格这么便宜，超级板正不卷边，都买它，买它

Google Translate:

Madam, madam, you are so beautiful! And the price is so cheap, the super board is not curled, everyone buys it, buys it.

Marco-o1-CoT:

fine, I need to translate the given Chinese sentence into English. The sentence is: "太太太太好看了！而且价格这么便宜，超级板正不卷边，都买它，买它"

First, let's break down the sentence to understand its components:

1. "太太太太好看了！" – This part translates to "**It's so beautiful!**" The repetition of "太太" (tài) emphasizes the beauty.
2. "而且价格这么便宜，" – This means "And it's so cheap," indicating the affordability.
3. "超级板正不卷边，" – This translates to "It's super straight and doesn't curl," describing the quality or appearance.
4. "都买它，买它" – This part means "Buy it, buy it," showing enthusiasm and a desire to purchase.

Now, considering the repetition of "买它" (buy it), it seems like the speaker is emphasizing their desire to buy the item multiple times, possibly due to excitement or emphasis.

Putting it all together, the translation should capture the enthusiasm and emphasis on both the beauty and affordability of the item, along with the repeated desire to purchase it.

Output:

It's so beautiful! And it's so cheap, super straight and doesn't curl. Buy it, buy it!

Figure 9 | Translation comparison of a colloquial expression “It's so beautiful! And it's so cheap, super straight and doesn't curl. Buy it, buy it!”.